Detecting Intermittent Steering Activity

Development of a Phase-detection Algorithm

Hugo M. Da Silva Peixoto de Aboim Chaves, Jasper A. Pauwelussen[c], Mark Mulder[a,b], Marinus M van Paassen[a], Riender Happee[b] & Max Mulder[a]

[a]Department of Aerospace Engineering, Delft University of Technology
[b]Department of Biomechanical Engineering, Delft University of Technology
Delft, The Netherlands
[c]Department of Perceptual & Cognitive Systems, TNO Behavioural & Societal Sciences
Soesterberg, The Netherlands
hugochaves_ist@hotmail.com; jasper.pauwelussen@tno.nl; mark.mulder@tudelft.nl; m.m.vanpaassen@tudelft.nl; r.happee@tudelft.nl; m.mulder@tudelft.nl

***Abstract*—Drivers usually maintain an error-neglecting control strategy (passive phase) in keeping their vehicle on the road, only to change to an error-correcting approach (active phase) when the vehicle state becomes inadequate. We developed an algorithm that is capable of detecting whether the driver is currently error-neglecting or error-correcting in straight lane keeping tasks. The development of this algorithm was part of a larger research project, DrivObs, that aims at creating an advanced driver observation tool. Performance of the algorithm in a straight lane driving task with lateral vehicle position perturbations was tested in a Monte Carlo simulation using Matlab/Simulink. Results show that the algorithm is capable of correctly detecting active or passive phase 90-95% of the time, depending on vehicle speed and algorithm settings.**

***Keywords*—Intermittent driver activity, driver observation, Monte Carlo simulation, human factors, driver support.**

I. Introduction

In 1988, [1] suggested that drivers usually maintain an error-neglecting control strategy (passive phase) in keeping their vehicle on the road, only to change to an error-correcting approach (active phase) when, in the opinion of the driver, the vehicle state becomes inadequate. Inadequacy was defined in terms of time-to-line-crossing (TLC). Drivers chose to commence error-correcting at approximately the same TLC≈1.4s for different driving speeds. Additionally, [1] reported that drivers attempted to switch to error-correcting when the vehicle state could still be smoothly improved, avoiding the need for harsh corrections. Given these observations on driver behavior, one may speculate on how distractions and other performance reducing factors can (negatively) impact the driver's switch to active phase.

In 1967, Senders [2] introduced the notion that drivers build an internal, predictive model of the vehicle behavior, and couple this with visual cues to control the vehicle. As such, when the driver's focus is not entirely dedicated to the driving task, the driver will tend to over-rely on this internal model of the vehicle without using the visual cues to update it as frequently as actually required. This in turn means that the internally predicted vehicle state will also tend to deviate more with respect to the real vehicle state compared to an attentive driving situation. This implies that it will be common, under these circumstances for the driver to assume that the vehicle state is acceptable, while in reality it is not. The main consequence is that the driver will tend to take longer to change to error-correcting behavior. From this it also follows that, because the driver is likely to have slower responses to changes in the vehicle state, these responses will need to be stronger, leading to more extreme steering corrections.

We developed an algorithm that is capable of detecting whether the driver is currently error-neglecting or error-correcting in straight lane keeping tasks. The development of this algorithm was part of a larger research project, *DrivObs*, that aims at creating advanced driver observation tools translating observed driving behavior into meaningful objective measures regarding the state of the driver and the resulting control strategies. DrivObs tools can be used off-line, to study driver behavior adaptation to proposed vehicle innovations [3]. This paper aims at on-line application, developing robust real-time driver state estimation such that the most adequate driver assistance can be given when needed. Detecting whether the driver is active or passive and whether the driver *should* be active or passive with regard to the current traffic context is important information with which it is possible to determine whether the driver is distracted or not and needs to be supported.

The work presented in this article details the development of an error-neglecting / error-correcting phase (subsequently called passive phase and active phase) detection algorithm. Performance of the algorithm was tested in a Monte Carlo simulation using Matlab/Simulink.

II. The Phase-Detection Algorithm

As described in the previous section, the driver may exhibit two distinct behaviors while performing a lane keeping task: a "standard" passive behavior and a corrective active behavior. The main differences between these phases are related to the 'steering energy' and the steering velocity, with the driver generally moving the steering wheel faster and more energetically during active phases. We defined the steering energy with the 'active factor' derived from the steering angle, as described below. A robust phase detection algorithm has been implemented using these 2 variables in what was called the *Trigger-Hold Mechanism*.

A. Definition of the Active Factor

The simplest energy related function is the root mean square (RMS) of a signal x with n number of samples. Two issues must be taken into account regarding the definition of the energy function though:

- The standard RMS function is centered around zero, which is coincidentally a good estimation of the 'neutral' steering position. However, a better approach is to estimate the neutral steering position using a forgetting saturated average of the steering angle. The advantage of using this approximation is two-fold: on the one hand, it overcomes any offset the steering angle sensor may have, and on the other hand it is a relevant reference in steady state following of (mild) curves.
- Because the goal of the steering energy function is to obtain an 'instantaneous' energy measure, instead of the average energy of the past steering history, it is important to include a forgetting factor that ensures a decreasing weight is given to older data. This forgetting parameter however should be large enough to ensure some temporal inertia, to be able to translate the amplitude of multiple brief steering actions into the level of activity during an active phase.

Taking these two considerations into account, we use the *ActiveFactor* defined in the following as a measure for the steering energy. The *ActiveFactor* of a steering wheel signal δ_H at the time instant n is given by:

$$ActiveFactor(n) = \sqrt{\frac{1-\lambda}{1-\lambda^{n+1}} \sum_{k=0}^{n} \lambda^{n-k} \left(\delta_H(k) - \delta_{H_{neutral}}(k) \right)^2} \quad (1)$$

where λ is the forgetting factor and $\delta_{H_{neutral}}$ is the neutral steering wheel position. The forgetting factor λ is chosen such that 95% of the total forgetting weight is given to the last 0.8 s of recorded samples.

B. Definition of the Steering Velocity

The steering velocity function is simply defined as the derivative of the steering angle signal: $\dot{\delta}_H$.

C. The Trigger-Hold Mechanism

The phase detection algorithm employs threshold based trigger and hold switches to enhance robustness, and to prevent spurious switches. The trigger-hold mechanism has two possible outputs being 0 (passive) and 1 (active steering).

The *ActiveFactor* has a relatively high temporal inertia to allow it to cope with the oscillations of the steering signal without being too affected by them, while $\dot{\delta}_H$ has an almost negligible temporal inertia, to ensure it reacts quickly to changes in the steering angle. As such, the $\dot{\delta}_H$ will tend to lead, with *ActiveFactor* lagging behind when detecting active and passive phases. Due to their distinct nature, we use the steering velocity to 'trigger' the active phase detection, and then use the *ActiveFactor* to 'hold' the active phase for as long as its value shows prolonged steering.

1) The Trigger: steering velocity. The goal of the steering velocity is to trigger the active phase as soon as a certain upper threshold, $Trigger_{max}$, is reached. Conversely, it could also be used to trigger the passive phase as soon as a certain lower threshold, $Trigger_{min}$, is crossed. However, a low steering velocity has no direct implications for detecting the driver's phase as this can happen in both active and passive phases. It is therefore desirable to make the lower boundary inaccessible in practice when $|\dot{\delta}_H|$ is small (for example giving $Trigger_{min}$ a negative value). The shape of the phase function in the transition region (0<Trigger<1) is directly related to the value of the *Hold* parameter and as such cannot yet be defined at this point.

2) The Hold: ActiveFactor. The purpose of the *Hold* parameter is to maintain the active or passive phase based on the *ActiveFactor* value. As such, when the *ActiveFactor* is equal or higher than an upper boundary $Hold_{max}$ (using a re-dimensioning process similar to the one for the trigger yields $Hold \geq 1$), the phase function should "hold" the active phase, i.e., always return the value one in the transition phase. Conversely, when the *ActiveFactor* is smaller or equal than a lower boundary $Hold_{min}$ (i.e., Hold $\leq$ 0), the phase function should not hold an active phase, by returning the value zero unless the trigger is 'pulled', i.e., in the transition phase. The shape of the phase function in the transition region can now be generalized for any *Hold* between zero and one by defining the hold axis.

3) Trigger-hold equations. To obtain a mathematical description of the trigger-hold algorithm, it is important to start by obtaining the position of the transition vertex. Inspection of the latter shows that the vertex is given by the point $V = [1 - Hold \; ; \; Hold]^T$. After obtaining the position of the vertex, the transition segments are given by eq's (2) and (3).

$$Phase(0 < Trigger \leq 1 - Hold) = \frac{Hold}{1 - Hold} \cdot Trigger \quad (2)$$

$$\begin{aligned} &Phase(1 - Hold < Trigger \leq 1) = \\ &\qquad Hold + \frac{1 - Hold}{Hold}\left[Trigger - (1 - Hold)\right] \end{aligned} \quad (3)$$

Coupling the transition equations with the trigger equations, the phase function, finally, is given by eq. (4). The algorithm interpretation based on the numerical value of *Phase* can be given by eq. (5).

$$Phase = \begin{cases} 0 & \forall\; Trigger \leq 0 \\ \frac{Hold}{1-Hold} \cdot Trigger & \forall\; 0 < Trigger \leq 1 - Hold \\ Hold + \frac{1-Hold}{Hold} \cdot [Trigger - (1 - Hold)] & \forall\; 1 - Hold < Trigger < 1 \\ 1 & \forall\; 1 \leq Trigger \end{cases} \quad (4)$$

$$\begin{cases} Passive & Phase \leq 0 + \delta_{Passive} \\ Maintain\ Previous\ Phase & 0 + \delta_{Passive} < Phase < 1 - \delta_{Active} \\ Active & 1 - \delta_{Active} \leq Phase \end{cases} \quad (5)$$

Where $\delta_{Passive}$ and δ_{Active} are the margins above zero and below one, respectively, for which a decision about the phase can already be made. In this work, they were chosen as $\delta_{Passive} = \delta_{Active} = 0.05$. To choose these values, it was considered that the phase function somewhat reflected the percentage of certainty for the phase to be active or passive, and as such this short margin should correspond to a very high certainty region already. It is important to remark that this choice only feebly affects the transition regions from active to passive and vice-versa (because when the algorithm is well calibrated it spends very little time in the transition zone).

The calibration of the Trigger-Hold algorithm corresponds, on the first level, to choosing adequate values of $Trigger_{min}$, $Trigger_{max}$, $Hold_{min}$ and $Hold_{max}$. On a second level, it also depends on the way that the *ActiveFactor* and the steering velocity are defined (namely on their forgetting factors).

III. Lane Keeping Monte-Carlo Simulations

To select optimal thresholds, and to assess the efficacy of the algorithm we created a straight lane keeping simulation with perturbations and distraction in MATLAB/Simulink. The simulation was executed as a Monte Carlo simulation to obtain a large variety of tested driving situations and algorithm parameters in order to investigate the quality and robustness of the algorithm. The simulated environment was run using a simple laptop computer.

A. MATLAB/Simulink model

The simulation model used for the Monte Carlo simulations was a simple compensatory tracking task with the driver tracking the center of a straight lane, see Figure 1.

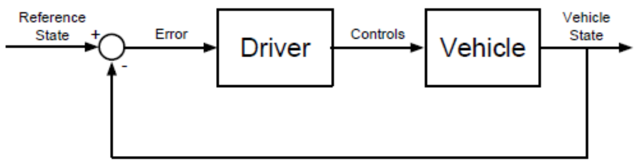


Figure 1. Block diagram of the general driver-vehicle closed loop simulation model as used in our Monte Carlo simulations.

The vehicle dynamics are based on the bicycle model, which is a 2 degrees of freedom model that can be written in the state-space form given by eq. (6).

$$\begin{bmatrix}\dot{V}_{y} \\ \dot{r}\end{bmatrix} = \begin{bmatrix} \frac{C_f + C_r}{m \cdot V} & \frac{a \cdot C_f - b \cdot C_r}{m \cdot V} - V \\ \frac{a \cdot C_f - b \cdot C_r}{J \cdot V} & \frac{a^2 \cdot C_f + b^2 C_r}{J \cdot V} \end{bmatrix} \begin{bmatrix} V_{y} \\ r \end{bmatrix} + \begin{bmatrix} -\frac{C_f}{m} \\ -\frac{a \cdot C_f}{J} \end{bmatrix} \left[\delta_{Wheels}\right] \quad (6)$$

where $\dot{V}_{y}$ is the lateral velocity in *m/s*; r is the yaw rate in *rad/s*; C_f and C_r are the respective front and rear cornering stiffnesses of respectively the front and rear axle in *N/rad*; m is the vehicle mass in *kg* and J is the vehicle moment of inertia in $kg{\cdot}m^2$; a and b are the respective distances from the front and rear axles to the center of gravity in *m*; V is the vehicle velocity in *m/s* and δ_{Wheels} is the steering angle of the front wheels in *rad*.

A driver model was applied consisting of three different blocks as shown in Figure 2. The position predictor block represents the internal model of the driver with regards to the vehicle dynamics; the reference path generator represents the driver's expectations with regards to the desired trajectory; and the control block represents the action dynamics that determine how the driver responds to perceived deviations from the desired reference trajectory.

The dynamics of the position predictor in state-space form were written as in eq. (7).

$$\left[y_{predict}\right] = \begin{bmatrix} 0 & 0 & 1 & V \cdot t_{predict} \end{bmatrix} \begin{bmatrix} V_{y} \\ r \\ y \\ \psi \end{bmatrix} \quad (7)$$

with y as the current lateral vehicle position in *m*; $y_{predict}$ as the predicted future lateral vehicle position in *m*; $t_{predict}$ the 'lookahead' time in *s* the driver uses to predict the future vehicle position; and ψ the yaw angle in *rad* ($r = \dot{\psi}$).

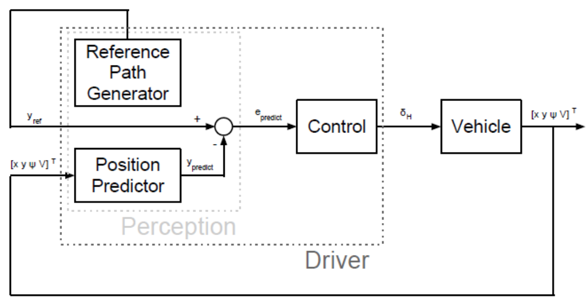


Figure 2. Extended block diagram of the driver-vehicle interaction where y_{ref} represents the driver's reference path; $y_{predict}$ the predicted lateral vehicle position; $e_{predict}$ the estimated future lateral error with respect to the lateral reference path; δ_H the steering wheel input to the vehicle; x the longitudinal position of the vehilce; y the lateral position of the vehicle; ψ the yaw angle of the vehicle; and V the speed of the vehicle.

Based on assumed driver equalization adjustment in the form of a low-frequency lead according to the Cross-over Model of McRuer et al. [4], [5], the control part of the driver was determined to have the transfer function as in (8).

$$H_{Control} = \frac{\delta_H}{e_{predict}} = \frac{SR \cdot \delta_{Wheels}}{y_{ref} - y_{predict}} = K \cdot (1 + \tau_{lead} \cdot s) \cdot e^{-\tau_{delay} s} \quad (8)$$

with δ_H the steering wheel angle in *rad*; *SR* the steering wheel to front wheel ratio; $e_{predict}$ the estimated future lateral error with respect to the lateral reference trajectory in *m*; K the lateral gain in *rad/m*; τ_{lead} the lead time in *s*; τ_{delay} the time delay in *s*; and s the Laplace operator. TABLE I shows the values used for the parameters in the simulation model.

Design and Procedure

1) Independent Variables. Three independent variables were used in this experiment. Vehicle speed was varied to see how that would influence the performance of the phase detection algorithm, V = {50, 80, 120}km/h. Furthermore, we were interested in effects of different calibrations of the phase detection algorithm, particularly the Hold, and used values of $Hold_{min}$ = {0, 1, 2, 4, 6, 8}deg and $Hold_{max}$ = {3, 6, 9, 12, 15}deg. Together, these independent variables yielded 3x6x5 = 90 different conditions that were tested. Each condition was repeated 100 times.

TABLE I. MONTE CARLO SIMULATION MODEL PARAMETERS

Parameter	Value	Source
vehicle		
C_f	95000 *N/rad*	[6]
C_r	85000 *N/rad*	[6]
m	1400 *kg*	[6]
J	3000 $kg{\cdot}m^2$	[6]
a	1.125 *m*	[6]
b	1.375 *m*	[6]
SR	15:1	[6]
driver		
$t_{predict}$	1 *s*	[7], [8]
$K_{lateral}$	0.13 *rad/m*	based on [5], ω_c = 3.3 rad/s
τ_{lead}	0.5 *s*	[5]
τ_{delay}	0.37 *s*	based on [5], ω_i = 2.0 rad/s
y_{ref}	*0 m*	-

ω_c is the cross-over frequency

ω_1 is the forcing function bandwidth

2) Simulation Procedure. The experiment consisted in keeping the simulated vehicle on the center of the simulated, infinite straight lane. The duration of each simulation was set to 1000 s in an attempt to assure asymptotic results for each performance parameter of the phase algorithm. The velocity was kept constant during each simulation run.

To ensure a rich sample of active and passive phases, a *'distraction switch'* was included in the Simulink model which randomly switched between an active or a passive simulated driver behavior. When the switch was on passive, this meant that there was zero driver corrective output for as long as the distraction switch was on, regardless of the vehicle state.

In order to elicit corrective steering actions, passive noise was added to the steering angle such that no single run would yield the exact same results. This passive noise was created from a sum of 4 sinusoids with random amplitude and random period. Details are given in TABLE II.

TABLE II. PASSIVE NOISE CHARACTERISTICS

Amplitude (deg)	Period (s)
0.4·(1+0.5·rand)	$45{\cdot}(1+0.3{\cdot}rand)^{-1}$
0.3·(1+0.5·rand)	$17{\cdot}(1+0.2{\cdot}rand)^{-1}$
0.04·(1+0.5·rand)	$7{\cdot}(1+0.1{\cdot}rand)^{-1}$
0.04·(1+0.5·rand)	$2{\cdot}(1+0.1{\cdot}rand)^{-1}$

rand represents a random, normally distributed number with zero mean and standard deviation one.

B. Evaluation Parameters

1) Match Percentage. The most simple and intuitive way to compare the phase distinction algorithm with a reference is by measuring the percentage of time they match. While this parameter provides a great overview on the general performance of the algorithm, it is incapable of providing a deeper understanding about how it is working, namely how well is the algorithm reacting to the transitions between phases and how well is it managing to hold the active and passive phases. As such, additional parameters will need to be devised to provide a deeper understanding of the algorithm's performance.

2) Activation and Deactivation Times. In order to better understand how the algorithm behaves in the transition regions, one possibility is to measure the average time difference between the algorithm and the reference for both the passive-active (activation) and active-passive (deactivation) transitions. While these parameters do not reflect the performance of the algorithm as a whole, they provide a very good understanding on how well the trigger and the hold are managing the activation and deactivation transitions, respectively.

3) Interruption and Overextension Ratios. To better evaluate how well the algorithm is able to hold the active phases (whichis one of the hardest parts to tune properly regarding the choice of the hold parameters), two concepts are introduced: interruptions and overextensions.

a) Interruptions: Interruptions are events where the reference phase remains active, but the algorithm is unable to hold the active phase, and as such yields a (usually, but not necessarily, short) passive output surrounded in between active outputs. Figure 3a illustrates this situation. Having a high interruption ratio (number of interruptions / number of active phases) can be considered a good indicator that the algorithm's hold is too weak, and therefore $Hold_{min}$ and $Hold_{max}$ should probably be decreased.

b) Overextensions: Overextensions are are the opposite of interruptions, corresponding to situations where the reference signal yields a passive phase in between two active phases, but the algorithm's hold is too strong, and as such overextends in this passive region, interconnecting the active phases, as illustrated in Figure 3b. Similarly to the interruption rate, the overextension rate is defined as the quotient between the number of overextensions and the number of active phases. Having a high overextension ratio will, as opposed to a high interruption ratio, indicate that the hold is probably too strong. The *Hold* parameters should, in that case, be increased to improve the performance of the algorithm.

C. Results

1) Match Percentage. Inspection of Figure 4a-c shows that the algorithm is capable of correct phase distinction around 90-95% of the time for a large zone of possible combinations of $Hold_{min}$ and $Hold_{max}$ and all three vehicle speeds. For 120 *km/h* the shape of the match percentage surface is slightly different, with a maximum value reached around $Hold_{min}$ = 4, and then a slight decrease for larger values of this parameter. It is also interesting to note that the match percentage appears to be much more affected by $Hold_{min}$ than by $Hold_{max}$.

2) Activation and Deactivation Times. The activation times for the different combinations of $Hold_{min}$ and $Hold_{max}$ were practically zero for each of the three speeds (data not shown). Deactivations times were found to be in the range between 0-0.5 *s* (data not shown) in a similarly dependent manner. $Hold_{max}$ and speed had little influence, while deactivation time decreased with increasing $Hold_{min}$.

3) Interruptions and Overextension Ratios

Figure 6a-c show that for the least amount of overextensions both $Hold_{min}$ and $Hold_{max}$ should both be as large as possible. As the velocity decreases the interruption ratio (Figure 5a-c) becomes much more affected by under-holding calibrations (this is particularly noticeable for $Hold_{min} \leq 4$). This effect seems to be non-linearly distributed with vehicle velocity. At 120 *km/h* overextension is not a problem for any of the $Hold_{min} - Hold_{max}$ combinations. At 80 *km/h*, however, a combination of low $Hold_{min}$ and low $Hold_{max}$ greatly increases the number of overextensions. A low $Hold_{min}$ has a noticeably more negative effect than $Hold_{max}$. However, at 50 *km/h* the increase of the number of overextensions for low $Hold_{min}$ and low $Hold_{max}$ combinations is reduced again. A low $Hold_{min}$ still has a more negative effect than a low $Hold_{max}$.

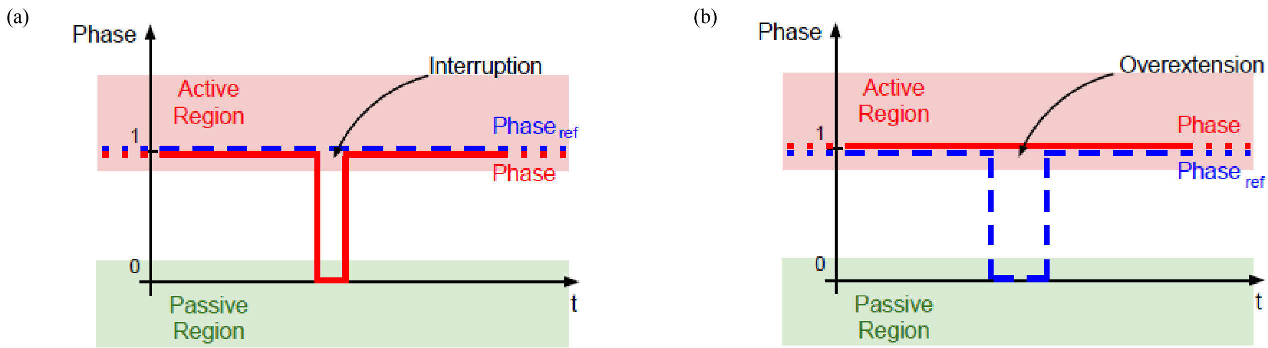


Figure 3. Schematic representation of an interruption (a) and an overextension (b) in the Phase(t) plot. The dashed (blue) lines represent the reference phase signal (i.e. what the driver is really doing), while the solid (red) lines represent the phase signal as estimated by the phase distinction algoritm.

IV. Discussion

We developed an algorithm to detect whether a driver currently is error-neglecting (passive phase) or error-correcting (active phase) with regards to a lane keeping task on a straight road. The first test of our algorithm was performed with a Monte Carlo simulation so that we could evaluate the two most important parameters of our algorithm, the minimum $Hold_{min}$ and the maximum $Hold_{max}$, over a range of different driving speeds. $Hold_{min}$ and $Hold_{max}$ affect the strength with which the algorithm 'holds' on to a current driver state (either active or passive phase). To get the best 'overall' performance out of the algorithm, the results of the Monte Carlo simulation suggest choosing a high $Hold_{min}$ and a high $Hold_{max}$ gives the best match over time with the actual behavior of the driver – approximately 90-95%, see Figure 4a-c. This result is also robust for the different speeds at which we tested the algorithm.

However, the smaller $Hold_{min}$, the more difficult it will be for the algorithm to switch into error-neglecting mode (passive phase). This is reflected in overextension results shown in Figure 6a-c. This figure indicates that the smaller $Hold_{min}$, the more likely the algorithm is to 'hold' on to the active phase, in other words, the overextensions increase. The smaller $Hold_{max}$, the more difficult it will be for the algorithm to switch into error-correcting mode (active phase). This also follows from Figures 6a-c. The opposite effect, however, can be detected for the interruptions. The larger $Hold_{min}$, the easier the algorithm will switch from active phase to passive phase (see Figure 5a-c). A large $Hold_{max}$ appears to be less relevant from inspection of Figure 5a-c.

Interestingly, the choice of $Hold_{max}$ and $Hold_{min}$ should be a compromise between few interruptions and more overextensions or more interruptions and fewer overextensions. Large $Hold_{min}$ (>4) are favourable for few overextensions, i.e. for reducing the number of undetected passive phases. Small $Hold_{min}$ (<4) are favourable for reducing the number of interruptions, i.e. for reducing the number of undetected active phases. In other words, when the active phase (error-correcting) is more of interest, a small $Hold_{min}$ is a good choice; when the passive phase (error-neglecting) is more of interest, a larger $Hold_{min}$ is a better choice. The choice of $Hold_{max}$ is less relevant for the overextensions or interruptions.

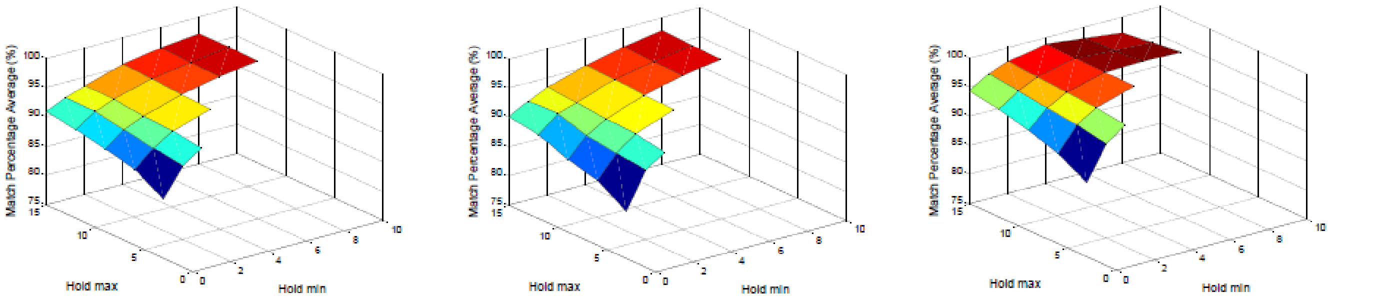
V. Conclusions

The driver error-neglecting / error-correcting distinction algorithm presented in this article was evaluated using a Monte Carlo simulation with MATLAB/Simulink. The results of the simulation show that overall performance of the algorithm is good, matching 90-95% of the driver's simulated behavior. Settings of the algorithm's *Hold* thresholds, .i.e. bounds that determine how quickly the algorithm switches between error-neglecting and error-correcting, need to be finely balanced, though. Depending on the driver state of interest, the lower threshold of *Hold*, $Hold_{min}$, should be chosen properly. A high $Hold_{min}$ (>4) is a good choice when error-neglecting behavior is of most interest; a low $Hold_{min}$ (<4) is a good choice when error-correcting behavior is of most interest.

Future work will evaluate the algorithm performance in more realistic driving settings in our fixed base driving simulator and ultimately in a real test-vehicle. Furthermore, the algorithm should be extended to be able to work in lane-keeping conditions other than on a straight lane.

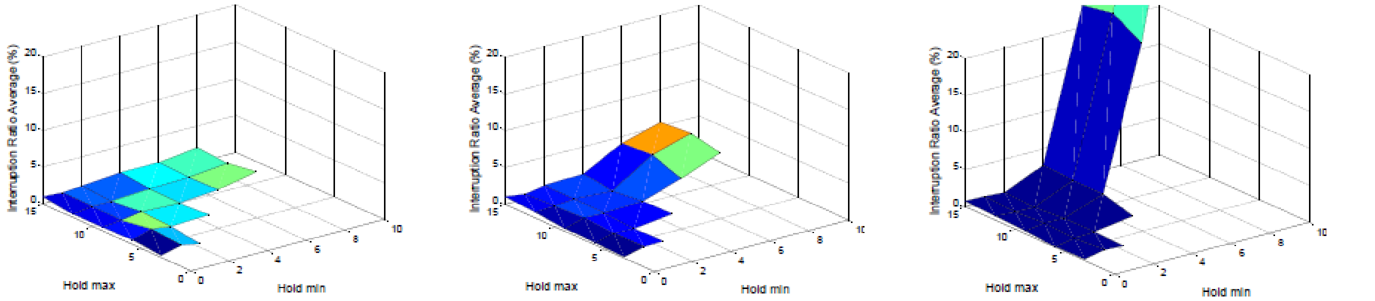
Acknowledgment

The work presented in this article was funded by the Dutch Ministry of Economic affairs, grant no. HTASI09004–E!5395 of the Driver Observation in Car Simulators (*DrivObs*) project (http://drivobs.3me.tudelft.nl).



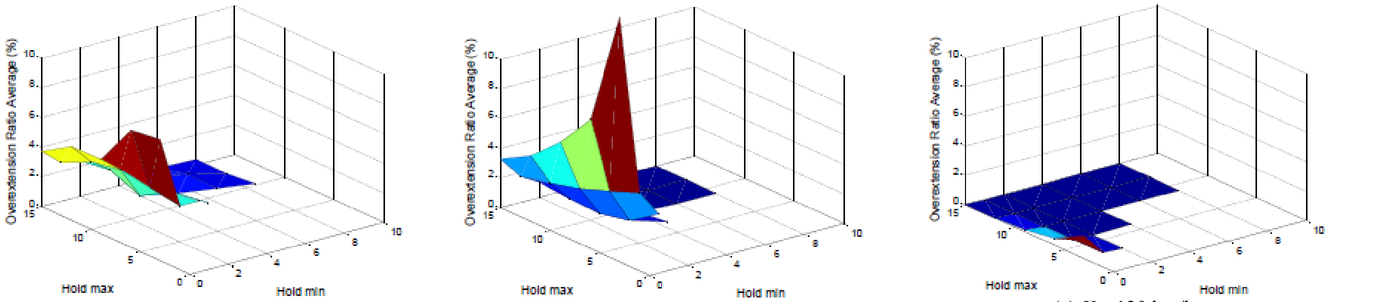
(a) $V = 50$ *km/h* (b) $V = 80$ km/h (c) $V = 120$ km/h

Figure 4. Match percentage averages after the 100 runs of the Monte Carlo simulations for each of the different combinations of V, $Hold_{min}$ and $Hold_{max}$.



(a) $V = 50$ *km/h* (b) $V = 80$ km/h (c) $V = 120$ km/h

Figure 5. Interruption ratio averages after on the 100 runs of the Monte Carlo simulations for each of the different combinations of V, $Hold_{min}$ and $Hold_{max}$.



(a) $V = 50$ *km/h* (b) $V = 80$ km/h (c) $V = 120$ km/h

Figure 6. Overextension ratio averages after the 100 runs of the Monte Carlo simulations for each of the different combinations of V, $Hold_{min}$ and $Hold_{max}$.

REFERENCES

[1] H. Godthelp, "The Limits of Path error-neglecting in Straight Lane Driving," *Ergonomics*, vol. 31, no. 4, pp. 609-619, 1988.

[2] J. W. Senders, A. B. Kirstofferson, W. H. Levison, C. W. Dietrich, and J. L. Ward, "The Attentional Demand of Automobile Driving," *Highway Research Record*, vol. 195, pp. 15-33, 1967.

[3] M. Saffarian, J. C. F. de Winter, R. Happee, "Enhancing driver car following performance through distance and acceleration display". In Press. IEEE Transactions on Systems Man & Cybernetics, Part A, 2012.

[4] D. T. McRuer, D. Graham, E. S. Krendel, and W. Reisener, "Human Pilot Dynamics in Compensatory Systems. AFFDL-TR-65-15.," Hawthorne, CA, USA, 1965.

[5] D. T. McRuer and H. R. Jex, "A Review of Quasi-Linear Pilot Models," *IEEE Transactions on Human Factors in Electronics*, vol. 8, no. 3, pp. 231-249, 1967.

[6] G. J. Heydinger, R. A. Bixel, W. R. Garrott, M. Pyne, J. G. Howe, and D. A. Guenther, "Measured vehicle inertial parameters-NHTSA's data through November 1998," *Transportation Research*, Society of Automotive Engineers, 400 Commonwealth Dr, Warrendale, PA, USA, 1999.

[7] S. Monsma and M. V. Oort, "Subjective Evaluation of Handling Behaviour related to Tyre Dependent Driver Parameters," *AVEC '10*, 574-579, 2010.

[8] J. Pauwelussen, "A basis for driver state estimation," *AVEC '10*, 622-627, 2010.